IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re the Application of: Gutmans, Andi	Application No.: 10/750719
Filed: 07/07/05	Art Unit: 2145
For: Efficient Handling of Download Requests	Examiner: Mirza, Adnan M.

DECLARATION UNDER 37 CFR 1.132

Mail Stop Amendment
Commissioner for Patents
PO Box 1450
Alexandria, VA 22313-1450

I, the undersigned, Zeev Suraski, hereby declare as follows:

1) I am an employee of the assignee hereof.

2) I have worked in the field of computer system programming and network communications for the past 13 years. I received a bachelor's degree in computer science from the Technion, Israel Institute of Technology in 1997. My experience in network programming and communications includes development of generic web tools, and distributed applications, and the supervision of research and development for the Assignee. I am a co-developer of the PHP Hypertext Preprocessor (PHP) and technologies for PHP.

3) I am familiar with the field of the document Gobin *et al.*, U.S. Patent No. 6,745,229 (Gobin), which was cited by the Examiner against the claims of U.S. Application No. 10/750,719 (hereinafter "the Application").

4) In the claimed invention a communication channel is established between a client and a server using a conventional socket. Independent claims 1, 10, and 18 of the Application are direct to a construction of a copy of this socket, processing the copy in a queue, and servicing a request of client using the copy. At paragraph [0036], the Specification states:

> the filename and the descriptor of the client connection socket that connects the server program 24 with a particular one of the clients 12 is transmitted to the download manager 28 . . . and is enqueued in the queue 32. Thus, a copy of the client connection socket is effectively transferred from the server program 24 to the download manager 28.

A person ordinarily skilled in the art would understand that copying a socket is equivalent to copying a file descriptor. Such copies can be achieved, for example, using the dup() functions that are documented in UNIX and Linux operating systems. The Linux programming manual explains that the calls dup and dup2 create a copy of a file descriptor. After successful return from calls to dup or dup2, the original descriptor and the copy may be used interchangeably. They share locks, file position pointers and flags. For example, if the file position is modified by using the Unix call lseek on one of the descriptors, the position is also changed for the other. Applicant discloses a more elaborate, but functionally equivalent copying technique in Listings 1 and 2.

The copy of the socket is then transferred to a queue for processing by a download manager. Simply stated, this operation constitutes a transfer of the socket connection from one process to another. This transfer does not require a proxy server, as explained in paragraph [0036] of the Specification.

Gobin discloses an entirely different technique that does not involve copying a socket connection. Instead, the original connection is established between the client and a proxy server in a DMZ (Fig. 2, DMZ 17, client 20 web servers 24). Then a new connection is created between the proxy server and a dispatch server 26, involving all the overhead of forming a new connection. The dispatch server then mediates the client request. Any needed information must be transferred from the original connection to the new connection via the proxy server, a technique known as "proxying information". The original connection is not actually copied as recited by independent claims 1, 10, and 18.

Furthermore, independent claims 1, 10, and 18 as amended recite maintaining an open connection with the client and the server via the copy of the socket while the request is awaiting service by the download manager in a queue. Gobin does not even disclose a queue of pending requests. In the current Official Action, the Examiner referred to (col. 3, lines 1-11), Contrary to maintaining an open connection, Gobin teaches that the connection between the dispatch server and the client is closed (col. 9, lines 13-25). The client is then required to iteratively re-establish a session, causing the server to reopen a socket connection. The client then polls the dispatch server to determine whether its pending request has been fulfilled.

In contrast, Applicant's claimed technique places the open copy of the socket in a queue with other sockets, The open connection with the client is maintained, which avoids the overhead of restarting a session with a new connection, and at the same time frees the server to deal with new client requests.

The differences between the steps followed according to Gobin and the claimed invention can be graphically appreciated by reference to Figures A and B, attached hereto.

Figure A illustrates the technique of Gobin, and has 4 phases:

Phase I: a first socket is opened between a client and a proxy server.

Phase II: a second socket is opened between the proxy server and a dispatch server, which distributes the request to other servers (not shown).

Phase III: The first socket is closed.

Phase IV: A third socket is opened between the client and a proxy server.

Phases III and IV alternate, resulting in an indefinite number of re-creations of the third socket until the result requested by the client is available. The requested information is then downloaded to the client via the last re-creation of the third socket.

Figure B illustrates the steps followed in the claimed invention and has three phases:

Phase I is similar to Gobin. A first socket is opened between the client and server. A second socket may be persistently maintained between the server and the download manager, or may be opened on demand.

Phase II: The first socket is duplicated by the server and the copy transferred to the download manager via the second socket.

Phase III: The original first socket is closed, thereby detaching the server from the client. The duplicated first socket remains open and now serves to connect the download manager with the client. Furthermore, the duplicated socket is placed in a queue with other sockets awaiting service by the download manager. The server, now unburdened from its connection with the client, is free to connect with other clients up to its limit of concurrent connections. The second socket is no longer relevant. It may be closed, or could remain open to deal with subsequent client requests. The duplicated first socket remains open until the request has been serviced, after which it may be closed.

Independent claims 1, 10, and 18 now include an element of enqueueing the copy of the socket in a download manager while maintaining an open connection with the client. The Examiner has asserted in paragraph 4 of the Official Action that this element is found in Gobin, citing (col. 3, lines 1 - 11), which mentions a a general capability to maintain customer sessions over the Internet. However, it is not reasonable to infer any particular implementation of this capability. There no explicit mention of queuing in Gobin. Indeed, a person ordinarily skilled in the field would not be able to arrive at the claimed implementation from reading Gobin without undue experimentation.

The Examiner considers that the following text from Gobin (col. 22, lines 26-28) discloses copying a socket (and enqueueing the socket and using the copy of the socket to download data).

> When the listener process has a data to pass to EDOCS000, EDOCS000 invokes a RETRIEVE command to get the data. EDOCS000 then performs a take socket and responds to the client by a write socket."

Gobin is here describing a flow diagram (Fig. 12) involving multiple steps of communications between the client and the server system, as well as a numerous steps that happen on the server side to provide the right response and perform the right action. All of this is disclosed with respect to the web server, dispatcher, and the proxy shown in Fig. 6 (see col. 22, lines 4-5). It is clear from the discussion that this paragraph deals with two entirely different sockets.

Moreover, there is no mention of socket queues. Gobin never mentions any changes in the way the server deals with the initial connection between the client and server as is claimed herein.

5) That it is my considered opinion that the claimed invention is a useful advance in the field that, to my knowledge, has not been realized in the art, and is not anticipated by Gobin.

6) I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and conjecture are thought to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application of any patent issued thereon.

Zeev Suraski
Citizen of Israel,

Date: May 18, 2008

Figure A (Gobin)

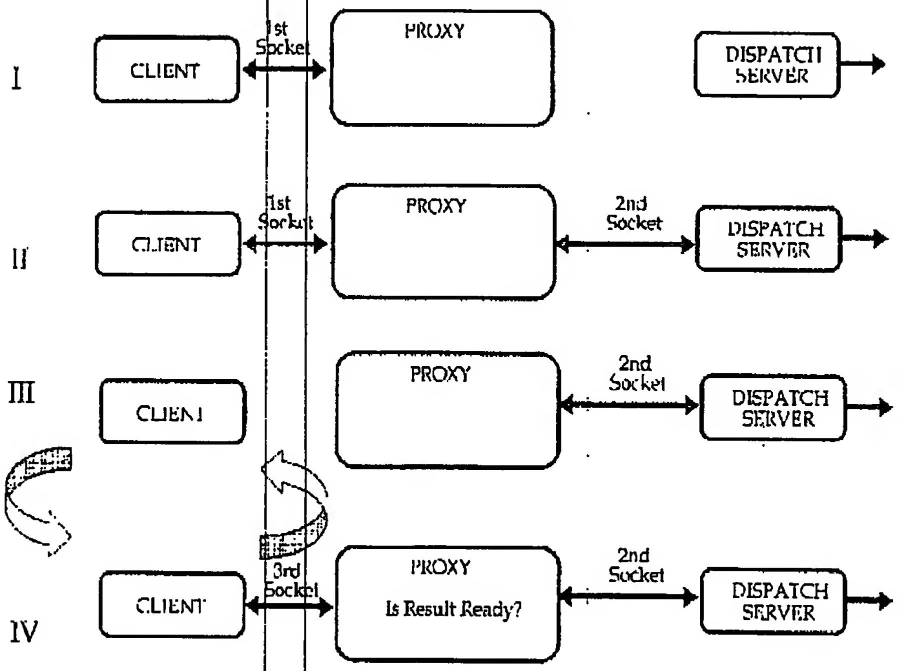
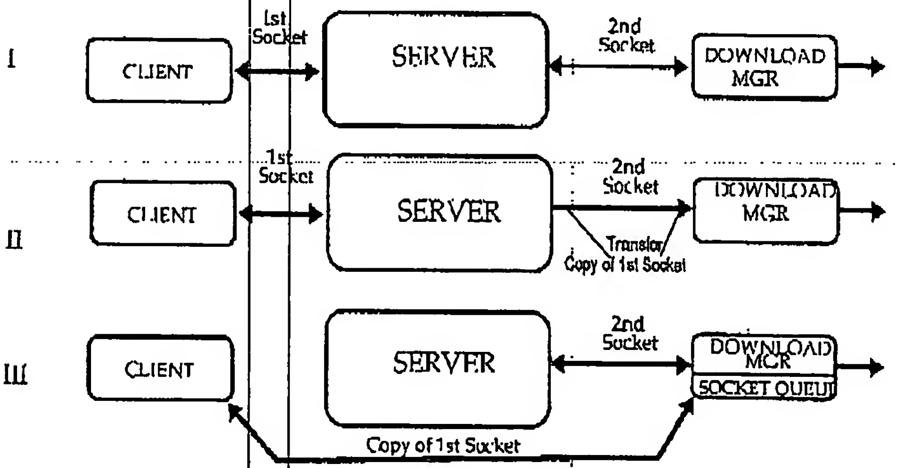


Figure B (Claimed Invention)


I
CLIENT
1st Socket
SERVER
2nd Socket
DOWNLOAD MGR
II
CLIENT
1st Socket
SERVER
2nd Socket
Transfer/ Copy of 1st Socket
DOWNLOAD MGR
III
CLIENT
SERVER
2nd Socket
DOWNLOAD MGR
SOCKET QUEUE
Copy of 1st Socket